DOCUMENT RESUME

ED 262 648 FL 015 324

AUTHOR Otto, Sue E. K.; Pusack, James P.
TITLE Tools for Creating Foreign Language CAI.
PUB DATE 85
NOTE 14p.; In: Meeting the Call for Excellence in the Foreign Language Classroom. Selected papers from the 1985 Central States Conference on the Teaching of Foreign Languages (FL 015 312).
PUB TYPE Information Analyses (070) -- Speeches/Conference Papers (150)

EDRS PRICE MF01 Plus Postage. PC Not Available from EDRS.
DESCRIPTORS *Authoring Aids (Programing); *Computer Assisted Instruction; *Courseware; Curriculum Development; *Microcomputers; Second Language Instruction; Teacher Developed Materials

ABSTRACT

Three basic kinds of tools exist for computer assisted instruction (CAI): templates, which are often highly user-friendly and require little investment of time but whose potential is limited; authoring systems, which offer a number of options for lesson design and greater variety in approach, but which require more sophisticated input from the teacher; and authoring languages, the most challenging kind of tool, which have built-in features for drilling, handling wrong answers, and scoring, and which give the author complete freedom to choose the logical arrangement of the lesson. The three kinds of tools overlap greatly, and the advantages and disadvantages are related to the teacher's goals and computing skills. Some of the fundamental features available in a number of forms are textual graphics, animation and simulation, and audio and video links. Increasingly useful tools are being created. The next step is for the tools to allow language teachers to work as experts in the field rather than as reluctant programmers. (MSE)

**********
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
**********

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

X The document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality.

• Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

"PERMISSION TO REPRODUCE THIS MATERIAL IN MICROFICHE ONLY HAS BEEN GRANTED BY
Natl. Textbook Co.
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

ED262648

FL015324

# 12
# Tools for Creating Foreign Language CAI

**Sue E. K. Otto and James P. Pusack**
**The University of Iowa, Iowa City**

## The Need for Tools

At all levels, language teachers are creators. Each drill, class handout, and test is a new creation. For the good language teacher, lesson-planning approaches choreography. Yet the spontaneous reaction to students' moods and needs often brings about the complete restructuring of a lesson as our continuing *commedia dell'arte* strives to intertwine culture and language structure in challenging new ways. Enter the computer. Suddenly the habitual spontaneity of lesson delivery must yield to the demands of logic, instructional design, yes-or-no decisions, and record-keeping. The conflict here is not between machines and illogical instructors. Experience shows that teachers skilled in the complexities of natural-language syntax have little difficulty in mastering the simple rules of programming languages. The conflict arises from the investment of time, energy, and expertise needed to make the primitive logical structures of computers *appear* to have the responsiveness and student-oriented flexibility that good language teaching demands.

Teachers have diverse perspectives on what can or should be accomplished by computer-assisted instruction (CAI) in foreign languages. Many hope to capture their own linguistic expertise and cultural knowledge in a computer program that can convey skills, facts, and insights to students in interesting, efficient ways. Or they may want to give students realistic communicative situations in which to acquire and use a foreign language. Since the ability of the computer to comprehend language and respond to reality will always be extremely limited, CAI must be viewed only as a preparation for real communication.

In order to exploit the computer as an effective extension of the language classroom, teachers must find ways of making it do their bidding. Prepackaged lessons will meet some of the profession's needs, but will never permit the personalized kinds of teaching that emerge from specific constellations of teachers and students. On the other hand, the job of programming one's own computer materials requires more dedicated effort than most language teachers can afford on a regular basis. A few teachers will engage in hours and months of nuts-and-bolts programming, but this can never be the model for the profession as a whole. For the typical language teacher who feels comfortable with computers, but who cannot become a full-time programmer, the answer will have to be the use of *authoring tools*. In computing, "to author" is a jargon verb that covers a broad range of creativity stretching from the typing of lists in a drill program to the elaborate design of animated simulations and tutorials. All of these activities have the same great vice and virtue: they are in some sense removed from the agonies of computer programming. An authoring tool gives a teacher power to create computer lessons at a "higher" level. As Alessi and Trollip have noted, this can be a mixed blessing:

> Compared to a programming language, most present-day authoring systems are not very flexible and force the program to conform to a relatively small number of alternatives. Not surprisingly, then, most programs developed on an authoring system look very much alike. However, because authoring systems are easy to use, require no knowledge of a programming language, and save time in producing a final product, we believe that authoring systems have an important future.[1]

Good language teaching, however, often demands far more than general-purpose authoring tools can currently deliver.

Authoring tools have been available on large computers for decades; many such tools have now been transferred to microcomputers, with mixed results. More of these tools will be developed in the coming years as microcomputers become familiar objects in our laboratories and classrooms. In order to make informed decisions concerning teachers' investments in the process of creating foreign language software, it is useful to survey the advantages and disadvantages of authoring tools for foreign language CAI. What can they offer teachers, and where will they probably fail to satisfy the demands of language pedagogy?

Each teaching situation involves compromises among a host of factors such as cost, effort, convenience, timeliness, and individual preferences and learning styles. Every attempt to select a textbook for a course confirms this rule. Computer programs for language teaching are no exception. CAI often disappoints us because it seems less, rather than more, flexible than the print medium. In selecting authoring tools, language teachers must be prepared to make difficult decisions about pedagogical approaches, graphic design, language processing, motivational elements, and the like. Typical authoring tools for microcomputers on the market today sacrifice some kinds of instr al power in order to provide capabilities in some other d Fairweather, who helped design what is possibly the ul authoring system available today, wisely observe: "The power of an authoring system can be measured by its functionality. The broader the range of functions the system can perform, the greater its power to solve any set of instructional problems."[2] No authoring tool can provide as many functions and features as a good teacher can imagine.

### Three Types of Tools

It is simple enough to demand that an authoring tool should be relatively friendly to the new user, but—as those who have learned to use more than one word-processor can testify—one's measure of friendliness changes over time. Here one encounters the classic "threshold-ceiling" problem. A novice user of any computer program wants results immediately, without confusing commands, complex rules of syntax, design decisions, or reference manuals. This demand is easily met by programs that continually display menus of design choices, prompt users for materials, check them for proper formats, and then generate a flawless lesson. Such *template* programs, as they are often called, usually produce a few basic types of interaction: multiple choice, true-false, and fill-in-the-blank. Vocabulary drill programs, for example, often take the form of templates. The amount of time invested is low compared to the amount of lesson material that results. After a few days or weeks, however, the advantages of this low threshold are exhausted. Instructors and students eventually want greater variety and the threshold then becomes a ceiling.

Two examples of highly imaginative and effective templates can be found in the programs STORYBOARD and CLOZEMASTER.[3] These

programs allow an author to use short foreign language texts for reading comprehension exercises. STORYBOARD requires students to guess all the words in a story based on a display where all the words have been replaced by asterisks. CLOZEMASTER generates a cloze procedure from the text, asking the students how often a word should be deleted. (The teacher can tell students what rate to choose.) Each new text is simply "plugged in" and used by the same program. The author's real task is to select texts which are short, challenging, and linguistically appropriate to a given group of students.

A more elaborate kind of template program is represented by *DASHER: An Answer Processor for Language Study*.[4] DASHER uses a system of graphic feedback to point out students' errors in basic grammar and writing exercises. Authors are prompted for instructions, examples, questions, and answers by the authoring program; the instructional program then handles the complete task of presenting items to the student and diagnosing wrong answers. Since full-sentence answers are possible, this template can rely on the creativity of the author to determine whether transformations, substitutions, synthetic sentences, sentence combining, fill-ins, scrambled sentences, dialogues, translations, vocabulary, or even dictation will be used. As with more powerful authoring tools, a number of features such as orthographic checking, messages to the student, availability of help and examples, and the symbols used to indicate errors can be altered by the author without recourse to programming.

DASHER is also typical of template-like tools in that one must accept the basic strategies employed as valid and effective pedagogical devices; otherwise, the tool can be of no value. This means that most templates are relatively easy to evaluate. Often, sample screen displays showing four or five student interactions—especially those where students make common errors—will suffice to establish whether a template is likely to prove useful in a particular teaching situation.

More complex template-like programs, known as *authoring systems*, offer a number of options in the design of lessons. One can often choose to circumvent the endless prompting that predefined templates use and make a number of decisions that drastically affect the flow of a lesson. Nevertheless, an authoring system still retains control over the lesson logic that determines how one frame will follow another. Authoring systems tend to become more complex with the passage of time. Using them may

very well become just as difficult as actually writing computer programs, although the final product may be much more impressive.

> A system can be powerful, productive and useable, and still be inappropriate for a given user if it requires such a high level of sophistication and training that the personnel available to the prospective user will be unable to take advantage of these features. Furthermore, if such sophistication can be gained only through very extensive training and long experience, the door to these powerful and productive authoring environments may well be closed for many users.[8]

Here a distinction must be made between the instructional features of the system—screen layout, answer evaluation, and the like—and the design of the system from the point of view of the author. Many systems may produce quite similar lessons, but offer radically different approaches to creating the lesson. Unfortunately, the only way to discover whether a tool is clumsy or efficient is to use it for several days or weeks.

One representative authoring system, the CAI Toolkit, serves to illustrate the kinds of features that can be provided without requiring the teacher to become a programmer.[9] The CAI Toolkit offers various kinds of spacing, highlighting, and character fonts; graphics tools for employing line drawings, rectangles, ellipses, stored shapes, animation, and color; music capabilities; timing of displays; analysis of numerical and textual answers, including compensation for misspelling and false word order; handling of correct, incorrect, neutral, and unexpected answers; branching based on student performance; and character set editing tools that can be used to display almost any foreign language writing system. Since this tool was designed for teaching languages, it can deal with the kinds of interactions needed for teaching structure and reading.

At the most challenging end of the scale one finds so-called *authoring languages*. An authoring language usually has built-in features especially designed for drilling, handling of wrong answers, and scoring. The hallmark of an authoring language is the author's complete ability to determine the logical arrangement of a lesson. No predetermined format must be used. Perhaps the best known authoring language for microcomputers is PILOT.[10] Like many other languages of this type, PILOT has a basic set of instructions which can be modified and combined into sequences to

produce programs of varying complexity. The instructions, most of which are mnemonic letter codes, direct the system to do a specific task: i.e., type some information, accept a student response, match a response with the answer, jump to another part of the lesson, or draw an image on the screen. This degree of control means that using an authoring language strongly resembles programming: algorithms must be designed and "debugged," often at the expense of great effort and time. With most authoring languages, manipulating linguistic structures in a systematic way can be a frustrating task.

These three kinds of tools—templates, authoring systems, and authoring languages—overlap greatly. The advantages and disadvantages of each depend on one's goals and computing skills. It is natural to ask whether an authoring language is always better than a template and therefore worth the investment of learning and frustration that goes with learning to program. The answer may well be negative if a specific template (or authoring system) can do an excellent or even an adequate job with far less teacher effort. A powerful prefabricated system that carries out worthwhile linguistic activities may actually perform tasks that no available authoring language can be made to perform. In this case, using an authoring language will produce inferior results with far greater effort. On the other hand, a well-designed authoring system may have built-in features for graphics, sound, or color that meet one's instructional needs precisely. While using these features takes time, the results are bound to involve creative uses of the computer in language teaching.

Even the most versatile of the three tools, the authoring language, remains less flexible than a true programming language and will stifle creativity in some sense. Sophisticated authoring systems avoid this predicament by permitting the author to "exit" temporarily to BASIC or PASCAL or even to another authoring tool. Since most language teachers cannot take advantage of such a feature, and since it is not yet possible to do so on most microcomputers, we will refrain from describing this utopia in detail.[8]

The numbers and types of good authoring systems available to language teachers are improving so rapidly that any listing would quickly become obsolete. This state of affairs gives rise to the kind of cautious optimism recently expressed as follows by Randall Jones.

> With the increasing availability of good authoring software language teachers can write CALI [computer-assisted language instruction] courseware in a variety of ways. They may choose to become adept at using a programming language, or they may prefer to spend more time on the lesson design and less on the program. The state of the art in CALI course authoring is now to the point that virtually any language teacher can write pedagogically sound and useful lessons. The effort required to transfer material from paper to the computer has been reduced significantly. It is important, however, that sufficient thought and planning be spent on the part of courseware authors, making certain that what ends up in the computer is worth the time spent using it.*

Jones rightly stresses the need for good lesson design. For new authors, the key must be a firm grip on pedagogical goals, combined with a keen eye for authoring tools with the most appropriate combination of features.

## Fundamental Features

Most of the authoring tools now available to language teachers were not designed to assist with language instruction. Many of them can handle nothing more than multiple-choice questions, often with little flexibility in the evaluation of students' responses. It is vital to examine authoring tools with care to see whether they meet the specific needs of the instructor. These needs will differ radically from course to course and from instructor to instructor. Some teachers will prefer to use the computer for drilling language structures, others will aim at diagnostic testing; some will use computers to motivate group activities such as conversation, and a few will hope to create dynamic reading texts or complex simulations of real-life situations. Across this diversity, a number of features are clearly fundamental to good foreign language teaching. In surveying these features, one should keep in mind that no single authoring tool can provide them all.

**Textual Graphics.** It is tempting to reject a whole series of authoring tools because they do not provide accents and umlauts. In general, it is difficult to work with tools that neglect this feature. Nevertheless, if a given tool has other powerful features not available elsewhere, out-of-hand rejection could be foolish. If a program will aid in the generation of complex adventure games, for example, this unique capability could more than make up for the extra effort involved in finding solutions to the problem of character sets. Software publishers have demonstrated interest in the

foreign-language market and can be persuaded to incorporate additional character sets into their products in later releases.

The shape of the characters on the screen is textual graphics at its most elementary. When we move to the display of charts, instructions, drill items, and reading passages, the capabilities afforded us by a good authoring tool become more difficult to define because the computer screen is so unlike the printed page. Although it is often smaller and thus holds less text than a typical book page, the screen can provide new ways of highlighting elements of text: boxes, inversing, color, and movement. Although most of these techniques are used in print, the computer adds the dimension of timing. A good authoring tool will allow an author to focus the learner's attention on words and sentences selectively, depending on student performance, and to illustrate linguistic structures by animating the relationship between parts of a sentence.

Lessons using these capabilities must be individually designed by the lesson author. Rigid tools such as templates have little room for creative freedom and must rely on predetermined graphics that usually cannot be changed. One can only evaluate the effectiveness of textual graphics in a template or authoring system and weigh the result with the tool's other features. A well-designed authoring tool, however, will put all of these elements at the disposal of the author in convenient ways. It should be possible to define windows on the screen which automatically display paragraphs with proper word breaks; the author should be able to erase text in one window without affecting other parts of the screen. Character sets of differing sizes and designs should be available, along with boldfacing, underlining, inverse video, and other focusing devices. Few authoring systems will have automatic means of moving words from one place to another on the screen, but it should be possible to create such effects within an authoring language. Color should be available for judicious use in highlighting relationships and establishing mnemonic links for tutorial purposes.

Our standard for good textual graphics have risen remarkably since the days when computer programs had all capital letters and programmers could blithely ignore punctuation. Without good visual design, students will struggle through CAI, groping for meaning as screens scroll past. An effective authoring tool will anticipate this need and teach a teacher to become a good visual designer.

**Language Processing.** For drill programs, tutorials, reading practice, and even adventure games, the capacity to handle natural language—usually the foreign language—is essential. Most good authoring tools can compensate for missing punctuation or false capitalization and spacing, many allow for a range of right and wrong answers to be specified by the author, and a few can generate effective diagnostic messages that lead the student toward a correct answer. These are fundamental features of a good authoring tool and should be investigated closely. The literal anticipation of wrong answers is a time-consuming process for the author and often involves logical complexities not immediately apparent. Any general-purpose authoring language should have what is called "partial answer processing" or keyword scanning to aid in this process.

Features which go beyond partial answer processing are not likely to be found in general-purpose tools, but language teachers themselves are beginning to create authoring programs that reach toward more sophisticated ways of interpreting students' responses. Programs that actually analyze a student's language are becoming available now, but they typically rely on tightly constrained linguistic environments in order to reduce the range of speech that must be understood. For this reason, true parsing programs are not likely to become components of authoring languages, although it may be possible to integrate them into carefully conceived templates.

Authoring tools available at the moment are especially ill-suited to generating sentence-level expressions. Many language teachers are strongly attracted to the prospect of using computers to generate exercises and test items based on vocabulary lists. The structures of elementary language study lend themselves to this approach and the need for seemingly endless practice makes computer-generated items a worthwhile goal. Nevertheless, most authoring tools have not anticipated this demand, or make it very difficult to accomplish the task.

In general, it can be said that complex tasks in the area of processing natural language—both production and analysis—require a great deal of computer power and expertise. Authoring tools are not designed for this purpose and many of these tasks are also difficult to accomplish in BASIC. Experiments in the simulation of language comprehension are usually carried out in artificial intelligence languages (LISP, Prolog) and have not yet reached the state where foreign language students can benefit widely

from them. When that time comes, there is no technological reason why this kind of power cannot be delivered through a new generation of authoring tools.

**Animation and Simulation.** Critics of the electronic workbook cite the superiority of those activities classified as "learner-controlled." A child's exploration of geometry through LOGO's turtle graphics is the classic example of such learner control: the student teaches the computer rather than the computer programming the student. One way of meeting the demand for learner control is to put authoring tools in the hands of students themselves. This solution has been used effectively with young language learners.

Computer simulations are another approach to learner control; they challenge the student to control the environment presented by the program. Simulations take a number of forms, including games, problem-solving exercises, and modeling or imitating of environments or activities. In a simulation students may be playing roles where they must guess or provide information to determine the outcome of the experience. When presented with stimuli in a context, the student must analyze them and respond; the program then reacts by creating a new situation based on the response.

Simulations present an appealing direction in foreign language courseware development. Certainly, the benefit offered by a well-constructed simulation is indisputable. It creates a highly interactive and motivational environment; the student learns actively, thereby increasing retention; learning takes place within a meaningful context; and repeated use broadens and reinforces the student's skills and comprehension without being boring. Impressive simulation programs exist for learning to fly a plane, practicing chemistry laboratory techniques, studying demographic trends, and even playing pinball. Unfortunately, there are no really good simulations available for foreign language study, much less authoring tools for creating them.

Simulations with foreign language themes typically have rudimentary game formats, which use insignificant segments of the target language and commonly allow incorrect usage. Moving ahead in producing simulations which can enhance language learning demands an arsenal of sophisticated authoring tools that are quite different from those offered by authoring languages and templates. The computer game construction sets

on the market which provide characters, locations, sound effects, graphics, and ways to glue them together into games constitute an excellent model for what effective simulation authoring tools must include. Besides tools for creating and orchestrating scenarios with high quality graphics and animation, there must be ways to handle intelligent, authentic language interaction. Communication with the program in the target language must produce appropriate branching to situations based on past and current student actions. Extensive, meaningful language interaction that simulates conversation poses a difficult problem, but the quality of language interaction in simulations will undoubtedly improve as progress is made in applying artificial intelligence to natural language processing.

**Diagnostic Control.** The traditional arguments in favor of using the computer as a tutor or drillmaster always list instant feedback on errors, selective branching to frames containing easier or harder material, based on student performance, tireless repetition without the drawbacks of human impatience, and presentation of items in random fashion. CAI authors must not limit their perspectives on effective computer-managed drill to the points on this list. As we explore the ways in which people assimilate information and the requirements for retention over time, new techniques are discovered for enhancing the learning process. For example, *optimization* is an important concept to consider in producing effective drill materials. The term *optimization* refers to the reinsertion of missed items at varying intervals in the "queue" or sequence of items yet to be presented. Simply put, it is a recycling process designed to encourage mastery. Typically, a missed item will be reinserted several times, the first time near the beginning of the queue, again approximately a third of the way down the queue, and again near the end. In this fashion missed items are drilled (and possibly reinserted) until the student masters them. In the case of a very long drill, there is an advantage to breaking the drill into a number of shorter sub-drills. Items not mastered within the sub-drill are then carried over. A useful related technique that can be used in drilling paired items is to re-drill not only the missed item, but, if the answer should indicate confusion with another pair, to do additional drill with that pair as well.

Recent emphasis on proficiency has renewed the demand for careful testing and record-keeping. The computer is an excellent vehicle for such efforts, particularly in the areas of grammar and reading. Systematic,

consistent testing is facilitated by the computer's vast potential for storing information, retrieving it according to prescribed categories, recording data about students' performance, and analyzing composite results. The records kept during practice or testing measure performance and guide progress in the course of acquiring knowledge. They also provide information that helps to evaluate the impact and effectiveness of the materials themselves. CAI projects in past years have relied on observation and intuitive feelings about the effectiveness of using the computer in various aspects of instruction, rather than conducting studies which gather hard data to support their assumptions. The trend now is toward keeping more records to find out how students perform, how they use the materials, and what is actually useful to them. Intelligent interpretation of this information will doubtless help refine existing programs and have a significant impact on the design of future ones. Potential authoring tools should therefore be carefully scrutinized for their record-keeping facilities.

**Audio and Video Links.** Some language teachers remain unconvinced by computer-assisted instruction composed of silent drills and computer-generated graphics that are disappointingly lacking in detail and realism. Even these skeptics find the possibility of linking a computer to a video or audio player an exciting prospect. Recordings can provide the authentic materials essential for sharpening comprehension skills. Yet passive exposure to such materials—just listening to or watching them without prior preparation and recourse to external aids—results in minimal gains and maximal frustration for the student. Effective strategies for meaningful, intensive work with recordings arise from the enhancements provided by the computer: precise control over playback of segments, help utilities (e.g., glossaries, paraphrases, comments), and comprehension checking. Unfortunately, the authoring systems marketed with computer-video interfaces generally support one type of interaction: multiple-choice comprehension checking. To limit the great potential of authentic video to a single lesson format is absurd. Satisfactory authoring tools must be flexible and sophisticated enough to handle a wide range of activities and formats, including creation of on-line aids for the viewer and of exercises based on cloze texts, prediction, simple description, sequencing of scenes, and management of in-class video activities. It is unlikely, however, that such advances will soon be achieved by commercial authoring tools.

### Shaping Future Tools

The determined author and the experienced programmer will shape their own tools to meet their creative needs. To do so, they may have to move beyond authoring tools or devise new ones. Meanwhile, CAI materials developed by language teachers will largely be shaped by the available tools. For a time at least, our methods tend to be those of the tools put into our hands, even though we eventually recognize their limitations. The problem becomes how to achieve productivity in CAI without sacrificing creativity. Those who rebel at the realization that not only their lessons, but also their students, are being wedged into templates will look for new approaches that facilitate learning rather than deliver teaching. Foreign language CAI will only provide nondirective access to linguistic and cultural knowledge if the next generation of authoring tools genuinely allows language teachers to work as experts in their own field, rather than as reluctant programmers.

#### Notes

1 Stephen M. Alessi and Stanley R. Trollip, *Computer-Based Instruction: Methods and Development* (Englewood Cliffs, NJ: Prentice-Hall, 1984), p. 33.

2 A. F. O'Neal and Peter G. Fairweather, "The Impact of Advanced Authoring Systems on CAI Productivity," paper presented at the Man and the Media Conference, Frankfurt, West Germany, June 12-15, 1984, p. 5.

3 Christopher Jones, *CLOZEMASTER: A Reading Skills Program* and *STORYBOARD: A Reading Skills Program* (London: WIDA Software, 1982).

4 James P. Pusack, *DASHER: An Answer Processor for Language Study* (Iowa City, IA: CONDUIT, 1983).

5 O'Neal and Fairweather, p. 16.

6 CAI Toolkit (Provo, UT: Softcat, 1984). See also Dennis Martin and Vardell Lines, "CAI Toolkit: A New Authoring System for Teaching Languages," *CALICO Journal* 1,iii(1983): pp. 43–45.

7 PILOT is available in various versions for most popular microcomputers. Although the language was not designed specifically for language teaching, many versions of PILOT do support foreign language character sets and interfaces for audio and video devices.

8 O'Neal and Fairweather label this a "hybrid" system and describe its implementation in the powerful WISE authoring system, distributed by WICAT Systems of Orem, Utah.

9 Randall L. Jones, "Authoring CALI Courseware," paper presented at the Man and the Media Conference, Frankfurt, West Germany, June 12-15, 1984, p. 19.

BEST COPY AVAILABLE